(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
24 October 2002 (24.10.2002)

PCT

(10) International Publication Number
WO 02/084956 A1

(51) International Patent Classification⁷: H04L 12/56, 29/06, H04Q 7/38

(21) International Application Number: PCT/SG01/00064

(22) International Filing Date: 16 April 2001 (16.04.2001)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant *(for all designated States except US)*: KENT RIDGE DIGITAL LABS [SG/SG]; 21 Heng Mui Keng Terrace, Singapore 119613 (SG).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: ZHOU, Luying [CN/SG]; Block 513, West Coast Road, #08-489, Singapore 120513 (SG). PILLAI, Radhakrishna, Pillai, Raghavan [IN/SG]; Block 218, Petir Road, #08-373, Singapore 670218 (SG). CHAN, Sek, Yuen, Pat [CA/SG]; Block 50, West Coast Crescent, #07-07 West Bay Condo, Singapore 128035 (SG).

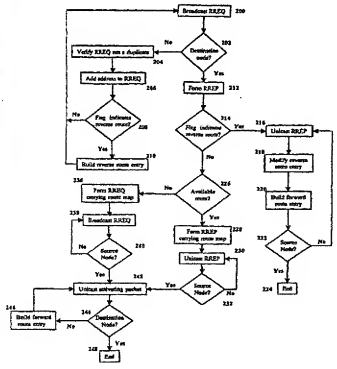
(74) Agent: GREENE-KELLY, James, Patrick; Lloyd Wise, Tanjong Pagar, P.O. Box 636, Singapore 910816 (SG).

(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, US, UZ, VN, YU, ZA, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE,

[Continued on next page]

(54) Title: A ROUTING PROTOCOL FOR GENERAL MOBILE AD HOC NETWORKS



(57) Abstract: An on-demand routing protocol based on both table look-up and route discovery mechanisms suitable for routing in mobile ad hoc networks with symmetric and/or asymmetric links. The routing protocol builds a route on demand, recording route information in a route discovery packet during the route discovery procedure. In the route discovery procedure, link state information relating to symmetric or asymmetric links is used to evaluate available reverse routes. When a route to the destination is discovered, a route reply message is sent back to the source via a reverse route, if one is available, or via some other route which needs to be found. After the forward route is established, data packets are routed according to route tables kept at nodes along the route, along carrying the entire route map. Route maintenance may be carried out by transmitting link failure information along previously established routes utilizing the failed link.

WO 02/084956 A1

IT, LU, MC, NL, PT, SE, TR). OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

A ROUTING PROTOCOL FOR GENERAL MOBILE AD HOC NETWORKS

TECHNICAL FIELD OF THE INVENTION

The present invention relates to mobile ad hoc networks, and more particularly to a routing protocol for use in mobile ad hoc networks with symmetric and/or asymmetric links which uses route discovery and table lookup mechanisms.

BACKGROUND OF THE INVENTION

The present invention relates to mobile ad hoc networks (MANETs). A mobile ad hoc network (MANET) is a collection of wireless mobile nodes dynamically forming a network without the use of any existing network infrastructure. In a MANET, there is no predetermined topology or central controller, and the MANET does not rely on a pre-existing fixed infrastructure, such as a wire-line backbone network or a base station. The responsibilities for organizing and controlling the network are distributed among the nodes themselves. The entire network is mobile, and the individual nodes are allowed to move at will relative to each other. Each mobile node in the MANET operates not only as a host but also as a router, as some pairs of nodes may not be able to communicate directly with each other and may require other nodes to relay packets.

MANETs typically have one or more of the following characteristics:

Mobility of nodes and dynamically changing topology. Nodes are free to move arbitrarily; thus, the network topology, which is typically multihop, may change randomly and rapidly at unpredictable times.

Asymmetric (unidirectional) links. An asymmetric link is caused by interference, differing radio or antenna capabilities, or adjustment of transmission and reception parameters such as power. An asymmetric link allows transmission between nodes in one direction only, while a symmetric link allows bidirectional transmission between the linked nodes.

Flat addressing. Nodes are unable to form a subnet or to have subnet addresses.

Energy-constrained operation. Some or all of the nodes in a MANET may rely on batteries for energy. For these nodes, power conservation is a critical design criterion.

Wireless vulnerabilities and limited physical security. Mobile wireless networks are generally more prone to information and physical security threats than fixed, hardwired nets.

MANETs can be used in military, rescue, and emergency applications, where either there is no other wireless communication infrastructure present or such infrastructure cannot be used because of security, cost, or safety reasons. MANETs can also operate as robust and inexpensive alternatives or enhancements of cell-based mobile network infrastructures.

Commercially, MANET technology may be used to extend the range of Wireless Local Area Network (WLAN) technology over multiple radio hops. Networks that cover areas of several square kilometers can be built from WLAN technologies such as HiperLAN and IEEE802.11. People and vehicles can thus be inter-networked in areas without a pre-existing communication infrastructure, or when the use of such infrastructure further requires extension by wireless means. On smaller scales, technologies such as Bluetooth™ (a radio interface using the 2.45GHz Industrial, Scientific, and Medical (ISM) frequency band, mainly used to offer short-range (~10 meters) data transmission via low-cost radio frequency modules and low power consumption) can be exploited in interesting ways to build embedded wireless networks. These networks would have a combination of static and mobile nodes, which could be fielded without cabling and with minimal pre-configuration. As computing and communication devices proliferate and the requirements of highly adaptive mobile networking technology increase, unforeseen uses of this technology are likely to emerge, particularly in the embedded systems and micro-networking fields.

Traditionally, end user devices, such as host computers or telephones, attach to distributed networks at fixed locations. As a consequence, they are assigned addresses based on their location in a fixed network-addressing hierarchy and often

assume an identity equivalent to their address. This identity-location equivalence greatly simplifies routing in these systems, as a user's location does not change.

In mobile ad hoc networks, on the other hand, the routing infrastructure can move along with the end devices. Thus, the infrastructure's routing topology can change, and the addressing within the topology can also change. Given this fundamental difference in the composition of the routing infrastructure in comparison to fixed infrastructure networks, much of the fixed infrastructure's control technology is no longer useful. The infrastructure's routing algorithms and, indeed, much of the networking suite must be reworked to function efficiently and effectively in this mobile environment.

The MANET working group (http://www.ietf.org/html.charters/manet-charter.html) in the Internet Engineering Task Force's (IETF–http://www.ietf.org/) Routing Area is chartered to provide improved mobile routing and interface definition standards for use within the Internet Protocol suite. In so doing, it hopes to lay the foundation for an open, flexible and extensible architecture for MANET technology. The routing protocols proposed in the MANET working group are developed in the network layer, because an inter-network layer routing solution is important for the following exemplary reasons: end user and application pressure for seamless internetworking will continue regardless of the underlying infrastructure (fixed or mobile); the "physical media independence" features of the network layer are important to support mobile routing through heterogeneous wireless fabrics, *i.e.* where different wireless technologies are used at the physical layer; definition of some common routing approaches and interface definitions provides future flexibility, and also improves the cost effectiveness of deployed systems.

Ad hoc On-Demand Distance Vector Routing (AODV) and Dynamic Source Routing (DSR) are two of the routing protocols which have been proposed in the IETF MANET working group. AODV and DSR both use an on-demand routing approach. The performance of on-demand routing approaches is superior to that of the table-driven routing approaches in ad hoc networks, as described in S.J. Lee et al, "A Simulation Study of Table-Driven and On-Demand Routing Protocols for Mobile

Ad Hoc Networks," IEEE Network, v.13, no. 4, pp. 48-54, July/August, 1999. With on-demand routing, when a route is needed, a routing protocol attempts to find a route for the current data communication session. On-demand routing does not require each node to continuously evaluate and maintain all the routes to every other node in the network, as required with table-driven routing, thus avoiding the need to frequently exchange state information, reducing the amount of update traffic, and conserving limited resources.

AODV is based on a distance vector routing mechanism and uses a route table to find the next hop in the route. AODV assumes symmetric links in the ad hoc network and hence cannot work properly in ad hoc networks having asymmetric links. DSR is based on a source routing mechanism and can work in ad hoc network having asymmetric links. However, DSR requires that the entire route map be carried with each data packet in order for the packet to reach its destination. Although DSR does not involve the route table lookup required by AODV, it nevertheless involves heavy routing overhead in each data packet.

Since the links in mobile ad hoc networks that connect neighboring nodes cannot be guaranteed to be symmetric, as mentioned in the foregoing; efficient routing protocols for networks having asymmetric links are needed.

Most existing ad hoc network routing protocols, such as AODV and Associativity-Based long-lived Routing (ABR), were designed for networks with symmetric links, and thus cannot be used effectively to establish routes in ad hoc networks having asymmetric links. Some existing ad hoc network routing protocols, for example DSR and Cluster Based Routing Protocol (CBRP), take into consideration the case of networks having asymmetric links, but they require each data packet to carry the entire route map to enable routing, resulting in excessively high overhead in every packet.

SUMMARY OF THE INVENTION

The present invention provides an on-demand routing protocol based on both table look-up and route discovery mechanisms which is suitable for routing in mobile ad hoc networks with symmetric and/or asymmetric links. The present invention includes a routing protocol for a mobile ad hoc network having a plurality of nodes interconnected by symmetric and/or asymmetric links, each node being associated uniquely with a corresponding route table. The protocol comprises discovering a route for a data packet from a first node to a second node, routing the data packet through the discovered route based on information in each route table along the discovered route, and maintaining the route.

In an embodiment of the routing protocol, the first node is a requesting source node and the second node is a neighbor node to the first node, and the discovering a route comprises checking a neighbor list.

In an embodiment of the routing protocol, the discovering a route comprises checking a neighbor list, broadcasting a first route request packet (RREQ), and receiving a route reply packet (RREP).

In an aspect of the embodiment, the first RREQ is received and modified by an intermediate node.

In an aspect of the embodiment, the intermediate node checks a status of a flag in the first RREQ and adds a reverse route entry to the route table corresponding to the intermediate node when the flag indicates a reverse route exists to the first node.

In an aspect of the embodiment, the intermediate node broadcasts the modified first RREQ.

In an aspect of the embodiment, the second node forms the RREP and unicasts the RREP to a next hop node along the reverse route.

In an aspect of the embodiment, the next hop node modifies the RREP and builds a forward route entry in the route table corresponding to the reverse route.

In an embodiment of the routing protocol, the discovering a route comprises checking a neighbor list, broadcasting a first route request packet (RREQ), and receiving a route reply packet (RREP) including discovered route information.

In an aspect of the embodiment, the first RREQ is received and modified by an intermediate node.

In an aspect of the embodiment, the intermediate node checks a status of a flag in the first RREQ and broadcasts the modified first RREQ when the status of the flag indicates a reverse route to the first node does not exist.

In an aspect of the embodiment, the second node forms a RREP using the modified first RREQ and unicasts the RREP to a next hop node along an available route.

In an aspect of the embodiment, the first node unicasts an activating data packet including the discovered route information to a node along the discovered route after receiving the RREP.

In an aspect of the embodiment, the node builds a forward route entry in the route table corresponding to the node.

In an embodiment of the routing protocol, the discovering a route comprises checking a neighbor list, broadcasting a first route request packet (RREQ), and receiving a second RREQ including discovered route information.

In an aspect of the embodiment, the first RREQ is received and modified by an intermediate node.

In an aspect of the embodiment, the intermediate node checks a status of a flag in the first RREQ and broadcasts the modified first RREQ when the status of the flag indicates a reverse route to the first node does not exist.

In an aspect of the embodiment, the second node forms the second RREQ using the modified first RREQ and broadcasts the second RREQ.

In an aspect of the embodiment, the first node unicasts an activating data packet including the discovered route information to a node along the discovered route after receiving the second RREQ.

In an aspect of the embodiment, the node builds a forward route entry in the route table corresponding to the node.

In an embodiment of the routing protocol, the information includes a route entry.

In an aspect of the embodiment, the route entry includes a timer and the route entry is erased when the timer expires.

In an aspect of the embodiment, the timer is reset when the route entry is used to route the data packet.

In an embodiment of the routing protocol, the maintaining the discovered route comprises determining that a next hop node is unreachable, transmitting link failure information to at least one upstream node, and erasing at least one route entry.

In an aspect of the embodiment, the transmitting link failure information comprises unicasting an error packet to the at least one upstream node.

In an aspect of the embodiment, the at least one upstream node modifies the error packet and unicasts the modified error packet.

In an aspect of the embodiment, the transmitting link failure information comprises broadcasting an error packet.

In an aspect of the embodiment, the at least one upstream node modifies the error packet and broadcasts the modified error packet.

In an embodiment or the routing protocol, the at least one upstream node erases the at least one route entry.

BRIEF DESCRIPTION OF THE DRAWINGS

Embodiments of the invention will now be described by way of example with reference to the drawings in which:

Figure 1a is a schematic diagram illustrating transmission of route request and route reply packets in an embodiment according to the present invention in an exemplary case where a reverse route exists.

Figure 1b is a schematic diagram illustrating transmission of route request and route reply packets in an embodiment according to the present invention in an exemplary case where a reverse route does not exist.

Figure 2 shows an exemplary embodiment of the routing protocol according to the present invention.

Figure 3 is a schematic diagram illustrating route maintenance in an embodiment according to the present invention.

Figure 4 shows an exemplary embodiment of the route maintenance mechanism according to the present invention.

DETAILED DESCRIPTION OF THE INVENTION

The present invention provides an on-demand routing protocol based on both table look-up and route discovery mechanisms which is suitable for routing in mobile ad hoc networks with symmetric and/or asymmetric links. The routing protocol of the present invention can exploit available symmetric link features of a network to improve efficiency. Data packets are routed along a discovered route based on information in route tables at nodes along the discovered route, reducing data packet overhead in comparison to source routing mechanisms.

The routing protocol according to the present invention builds a route on demand and works efficiently in ad hoc networks with symmetric and/or asymmetric links between nodes. Route information is recorded in a route discovery packet during a route discovery procedure. In the route discovery procedure, link state information relating to symmetric or asymmetric links is used to evaluate available reverse routes. When a route discovery packet reaches the destination node, a route reply message may be sent back to a source node via a reverse route, if one is available, or via some other route which needs to be found. After the route is established, data packets are routed according to route tables kept at the nodes along the route. As used herein, "reverse route" indicates a route from the destination node to the source node of a source, destination node pair, and "forward route" indicates a route from the source node to the destination node of the node pair. In addition, the

format of reverse and forward route entries in the route tables is identical. Data packets do not need to carry the entire route map with them as is the case with the DSR protocol. Route maintenance may be carried out by transmitting link failure information along previously established routes that incorporate the failed link.

Each node in the network maintains a neighbor list that stores link state information indicating whether its neighbors can be heard and/or are reachable, that is whether the node can receive transmissions from or send transmissions to its neighbor nodes. Assuming that a node has N neighbors, the format of its neighbor list may be as shown, for example, in Table I:

Table I

Node	Status
Node_1	Stat_1
...	
Node_N	Stat_N

In Table I, *Node_x* indicates the Internet Protocol (IP) address of neighbor node *x*, where 1 x N. *Stat_x* provides link state information, *i.e.* the status, of the link connecting the node and a neighbor node *x*. *Stat_x* can take on one of three values: forward, reverse, or bidirectional. Forward status indicates that the neighbor node *x* can receive packets sent by the node. Reverse status indicates that packets sent by the neighbor node *x* can be received by the node. Bidirectional status indicates that the neighbor node *x* can receive packets sent by the node and vice versa. If the status is bidirectional, then the link connecting the node to its neighbor node *x* is symmetric. Otherwise the link is asymmetric.

The neighbor list can be built and updated through various mechanisms such as promiscuous listening, beacon messaging, explicit 3-way handshaking or through periodic Hello message exchanges, as is known in the art.

In order for a data packet to be sent from a source node to a destination node, a transmission route must exist. If a node, by checking its neighbor list, determines that the destination node is a neighbor node, then it can send the data packet to the

destination node directly. If the destination node is not a neighbor, then a route must be discovered before the data packet can be sent to the destination node.

The details of the route discovery and data packet routing mechanisms according to the present invention will now be described.

In order to discover a route to the destination node, the source node transmits a route discovery packet, referred to as a route request packet (RREQ), by broadcasting to its neighbor nodes. This RREQ will be rebroadcast again throughout the network by intermediate nodes and finally be received by the destination node.

In one embodiment, the format of the RREQ is as shown in (1) below:

Type	Dest_addr	Src_addr	Req_id	Flag	Route_info	Option

(1)

In the RREQ (1), the various fields may be described as follows: *Type* identifies the packet (*e.g.*, in this case Type = RREQ); *Dest_addr* indicates the IP address of the requested destination node; *Src_addr* indicates the IP address of the requesting (source) node; *Req_id* is a sequence number generated by the source node to prevent the processing of duplicate RREQs received by intermediate nodes through different paths; *Flag* indicates whether a node along a route can build a reverse route to the requesting source node. If all links connecting intermediate nodes along the route from the requesting source node to the node are symmetric, then a reverse route can be built from the node to the requesting source node along the route, in a direction from the node to the source node; *Route_info* is a list of IP addresses of intermediate nodes along the RREQ propagation route, in the order in which the intermediate nodes receive the RREQ; the *Option* field may contain discovered route map information for the case where the requested destination node needs to discover a route to the requesting source in order to send a reply packet, as described further below.

When a node receives a RREQ, it first checks the *Src_addr* and *Req_id* values to avoid rebroadcasting duplicate RREQs. The node records *Src_addr* and *Req_id* values for each RREQ for the purpose of comparing RREQs. A duplicate

RREQ (a RREQ with *Src_addr* and *Req_id* values identical to a previously received RREQ) is discarded by the node.

If a node is not the requested destination node, then it modifies the RREQ appropriately, as described in the following. The node adds its own address to the sequence of addresses *Route_info* in the RREQ. Furthermore, the node checks the *Flag* status. If *Flag* has been set to 1, which means that no reverse route can be built from the upstream node (as determined from the last address in the *Route_info* field in the packet) to the requesting source node, then the *Flag* status is left unchanged. If *Flag* has been set to 0 and the link connecting the node and the upstream node is symmetric, signifying that a reverse route can be built from the node to the requesting source node, then the *Flag* status is left unchanged. Otherwise, if *Flag* has been set to 0 and the link connecting the node and the upstream node is asymmetric, signifying that the node can receive directly a packet sent by the upstream node while the upstream node cannot receive directly a packet sent by the node (*i.e.*, the packet must travel from the node to the upstream node by a circuitous route), then the node will set *Flag* to 1. If *Flag* remains at 0, the node adds a reverse route entry to its route table.

Each node in the network is associated with its own route table. An exemplary form of a route table is shown in Table II, where it is understood that the empty cells in the table are filled with valid values. Each row in the route table is referred to hereinafter as a "route entry" or simply "entry."

Table II

Dest_addr	Nxt_hp_addr	Up_str_addr	Hop_cnt	Time_stmp

The various fields in Table II, indicated by the headings in the first row of Table II, contain the following data. A reverse route is built from a current node to the requesting source node. *Dest_addr* holds an address of the destination node to

which the packet will be sent ultimately. Here, it is filled with the address of the requesting source node (*Src_addr* in the corresponding RREQ). *Nxt_hp_addr* holds an address of the next hop node, to which the packet will be sent immediately. Here, *Nxt_hp_addr* contains the address of the upstream node, obtained from *Route_info*. *Up_str_addr* holds an address of the upstream node from which the packet is received. *Up_str_addr* can be extended to form a list containing more than one upstream node address. Here, *Up_str_addr* holds the address of the node to which the RREQ may be sent. (This field is filled later, when a corresponding forward route is built, and used in route maintenance in order for the node to be able to inform its upstream node of a failed link.) *Hop_cnt* holds a number of hops from the current node to the destination node. Here, *Hop_cnt* contains the hop count from the current node to the requesting source node, derived from *Route_info*. *Time_stmp* holds a time value. Here, *Time_stmp* contains the current time. Because of node mobility, a discovered route may become stale after some time. Thus, each entry or row in a route table is kept only for a short period of time in order to prevent a stale route from being used. The *Time_stmp* in each entry is periodically compared with the current time, and if the difference is more than a given value, then the corresponding entry is deleted from the route table.

After modifying the RREQ by adding the current node address in the *Route_info* field and adjusting the value of *Flag* in the RREQ, the current node broadcasts the modified RREQ.

If a node is the requested destination node, the node forms a route reply packet (RREP) and sends it back to the requesting source node. If a reverse route from the requested destination node to the requesting source node is available, the RREP will be sent along the reverse route. Otherwise, the RREP will be sent along an available route to the requesting source node, or the destination node will find a new route through the route discovery procedure and piggyback the discovered route information on another RREQ, as described in greater detail hereinafter. Figure 1a illustrates the propagation of the RREP and RREQs in an exemplary case where a reverse route exists, *i.e.*, when Flag = 0: the RREP travels along the same path that

the RREQ traversed. Figure 1b illustrates the propagation of the RREP and RREQ in an exemplary case where a reverse route does not exist, *i.e.*, when Flag = 1: the RREP cannot travel along the path that the RREQ traversed, and a different route must be used. In Figures 1a and 1b, a circle indicates a node in a mobile ad hoc network, a dashed line or dashed arrow indicates a link between two nodes, and a solid arrow represents a RREP and/or RREQ.

Referring to Figure 1a, a route is established through the propagation of a RREQ **110** from source node **100** through intermediate node **102** to destination node **101**. Since a reverse route exists, the destination node **101** unicasts a RREP **120** along the reverse route (through node **102**) back to the source node **100**. Each node along the reverse route, *e.g.* **102**, examines the RREP **120**, completes the reverse route entry by filling the immediate sender's address in the *Up_str_addr* field in the reverse route entry in the route table, and builds a forward route entry in the route table leading to the requested destination node. Thus, the RREP **120** does not need to carry the entire route map.

An exemplary format of a RREP according to the present invention is shown in (2) below:

Type	Dest_addr	Src_addr	Hop_cnt

(2)

The various fields in the RREP contain the following data: *Dest_addr*, the address of the requesting source node *(Src_addr* in the corresponding RREQ); *Src_addr*, the address of the requested destination node (*Dest_addr* in the corresponding RREQ); and *Hop_cnt*, the hop count from the requested destination node, initially set to zero at the requested destination address.

The RREP is unicast to the node, *e.g.* **102** in Fig. 1a, having the address indicated in *Nxt_hp_addr* field in the reverse route entry. When the next hop node (**102** in this example) receives the RREP, it gets the immediate sender's address (the address of node **101**) and places it in the *Up_str_addr* field in its corresponding reverse route entry. This *Up_str_addr* value will be used in the route maintenance

procedure described below to inform upstream nodes of unavailable link status. The next hop node (102 in this example) also builds a forward route entry in its route table leading to the requested destination node (node 101, in this example). The forward route entry has the same format as a reverse route entry, as shown, for example, in Table III below, where it is understood that the empty cells in the table are to be filled with valid values.

Table III

Dest_addr	Nxt_hp_addr	Up_str_addr	Hop_cnt	Time_stmp

The various fields in Table III, indicated by the headings in the first row thereof, contain the following data: *Dest_addr*, the address of the requested destination node (*Src_addr* in the RREP); *Nxt_hp_addr*, the address of the previous node (from which the RREP was received); *Up_str_addr*, the *Nxt_hp_addr* in the corresponding reverse route entry; *Hop_cnt*, the hop count, obtained from the RREP; and *Time_stmp*, the current time.

A node which receives a RREP will modify the RREP by increasing the value of *Hop_cnt* by one and re-send the RREP to its next hop node along the reverse route. In the example above, node **102** receives RREP **120**, which has *Hop_cnt* = 0, set by destination node **101**. Node **102** increases the value of *Hop_cnt* to 1 and re-sends RREP **120** to node **100**, its next hop node along the reverse route. When the RREP finally reaches the requesting source node along the reverse route (node **100** in the example above), a forward route from the requesting source node to the requested destination node is discovered accordingly.

In the case where a reverse route does not exist, for example as shown in Figure 1b, we consider two scenarios. In the first scenario, another existing route is available and may be used. In the second scenario, there is no available route, and the previously discovered route information can be piggybacked on another RREQ to

the requesting source node. Turning to the first scenario, where another route is known and available, for example from requested destination node **151** to source node **150** through node **153**, node **151** extracts *Route_info* from the RREQ **160**, forms a RREP **170** in response to the RREQ, and unicasts it to the requesting source node along the available route through **153** to **150**. In contrast to the case discussed previously and shown in Figure 1a, in this case, the RREP does carry discovered route information. The RREP **170** in this case may have the format shown in (3) below, where the field *Extracted Route_info* contains information extracted from the *Route_info* field of the RREQ **160**:

Type	Dest_addr	Src_addr	Extracted Route_info

(3)

In the second scenario, where an alternate route is not known, the destination node **151** inserts the appropriate *Route_info* value into the *Option* field in a RREQ **170** and sends it to the requesting source node **150** by broadcasting, that is, the discovered route map "piggybacks" on the RREQ **170**. The format of the RREQ **170** in this instance is the same as in (1) and is shown in (4) below:

Type	Dest_addr	Src_addr	Req_id	Flag	Route_info	Option

(4)

When the requesting source node **150** retrieves the information from the *Option* field in RREQ **170** having the format (4), it knows the entire route map leading to the requested destination node **151**.

In the first and second scenarios described above, after the RREP or RREQ is received by the requesting source node, data packets can be routed to the destination node by using the Loose Source and Record Route option in IPv4 or the route header option in IPv6. (The Loose Source and Record Route option in IPv4 provides a means for the source of a packet to supply routing information to be used by intermediate nodes in forwarding the packet to the destination and to record the route information. The route header option in IPv6 allows the source of a packet to list the

intermediate nodes to be "visited" on the way to a packet's destination.) In order to avoid carrying an entire route map with each data packet, the first data packet sent by the source node serves as an activating route packet to enable the nodes along the discovered route to build route entries in their route tables. This is in contrast to the scenario in which a reverse route exists (Figure 1a), wherein, as previously described, the forward route is built along the discovered route as the RREP propagates from the requested destination node to the source node, and consequently no activating packet is required. Based on the routing information in the Loose Source and Route Record option or the route header option in the data packet, each intermediate node along the route, *e.g.* **152** in Figure 1b, is able to build a forward route entry in its route table. The fields in the forward route entry, such as *Up_str_addr*, *Nxt_hp_addr* and *Hp_cnt*, can be filled by examining the route information carried in the option header of the data packet. Thus, a forward route from the requesting source node, *e.g.* **150**, through **152** to the requested destination node **151** can be established.

If more than one RREQ is received at the destination node, one of the routes in the RREQs may be chosen based on certain metrics (such as number of hops, signal strength, existence of reverse route, etc.).

When route information is set in the route tables of the nodes along a route, subsequent data packets are routed to the destination node based on the data in the route tables held by each node without the need to carry the entire route map in the subsequent data packets.

When a first node needs to communicate with a second (destination) node, it first checks whether a route to the second node exists in its routing table. If a route does not exist, the first node broadcasts a RREQ to find a route to the destination node. If a route does exist from the first node to the destination node, the first node sends a data packet to the intended destination node based on route table information held by intermediate nodes. Each intermediate node along the route forwards the packet based on its route table.

Figure 2 provides an overview of an exemplary embodiment of the protocol according to the present invention. At 200, the requesting source node broadcasts a first RREQ. At 202, a node receiving the first RREQ determines whether it is the requested destination node. If the receiving node is not the destination node, at 204 the receiving node verifies that the received first RREQ is not a duplicate, as described previously. At 206 the receiving node adds its own address to the first RREQ and at 208 checks the flag status to determine whether a reverse route to the requesting source node exists. If not, the receiving node broadcasts the modified first RREQ. If yes, the receiving node at 210 builds a reverse route entry in its route table and broadcasts the modified first RREQ.

If the receiving node is the requested destination node, at 212 the receiving node forms a RREP and at 214 checks the flag status in the previously received first RREQ to determine whether a reverse route exists to the requesting source node. If yes, at 216 the destination node unicasts the RREP to its next hop node along the reverse route. At 218 the next hop node completes the reverse route entry in its route table, as described previously, and builds a forward route entry in its route table at 220. At 222 the next hop node determines whether it is the requesting source node. If yes, route discovery is complete (224). If not, the next hop node unicasts the modified RREP to the next hop node along the reverse route.

If the receiving node is not the requested destination node, at 226 the destination node determines whether an available route exists the requesting source node. If yes, the destination node forms a RREP containing information regarding the discovered route, as described previously (228) and unicasts the data packet to a node along the available route (230). At 232, if the node receiving the data packet is not the requesting source node, the receiving node unicasts the RREP to another node along the route. If the requesting source node receives this RREP, it unicasts an activating data packet (242). At 244, a node receiving this activating packet determines whether it is the requested destination node. If the receiving node is not the destination node, at 246 the receiving node builds a forward route entry in its

route table and unicasts the activating packet. If the receiving node is the destination node, route discovery is complete (248).

If at 226 the destination node determines that an available route to the requesting source node does not exist, the destination node forms a second RREQ (236) containing information regarding the discovered route, as described previously, and broadcasts the second RREQ (238). At 240, a node receiving the second RREQ determines whether it is the requesting source node by examining the RREQ, as described previously. If the receiving node is not the requesting source node, the receiving node rebroadcasts the second RREQ. If the requesting source node receives the second RREQ, it unicasts an activating data packet (242), which is treated as described in the preceeding paragraph and shown in Figure 2. A route maintenance mechanism according to the present invention will now be described. When a current node notices, using, for example, one or more of the mechanisms listed previously for building up the neighbor list, that the next hop node along a route is unreachable, it must notify the upstream nodes using the route to update the route information held by each intermediate node in a timely manner. In this circumstance, the current node sends an error packet to the upstream node indicated by the corresponding route entry in its route table to inform the upstream node that the destination node is unreachable. Two different ways to send the error packet are considered below.

First, if there exists a reverse route or other available route to the upstream node from the current node, then the current node unicasts an error packet to the upstream node. The format of the error packet may be as shown in (5) below:

Type	Dest_addr	Src_addr	Route_Dest_addr

(5)

The various fields in the error packet having the format shown in (5) contain the following data: *Dest_addr*, the address of the upstream node in the relevant route; *Src_addr*, the address of the current node, which is forming the error packet;

Route_Dest_addr, the address of the destination node in the relevant route; and *Type*, an indication that the packet is a unicast route error packet.

When the upstream node receives the error packet, it erases the route entry (*i.e.* row) in its route table where the values in the fields *Nxt_hp_addr* and *Dest_addr* in the route entry match the values in the *Src_addr* and *Route_Dest_addr* fields in the error packet, respectively, and sends a similar error packet to its next upstream node.

The following example illustrates the functioning of route maintenance according to the present invention, in the exemplary case where two routes have been discovered in the mobile ad hoc network. Figure 3 diagrammatically illustrates the case where source node **300** has discovered a route to destination node **306** consisting of, in sequence, nodes **300, 301, 305, 302,** and **306**, while source node **303** has discovered a route to the same destination node **306** consisting of nodes **303, 304, 305, 302** and **306**. In Figure 3, a circle indicates a node in a mobile ad hoc network, a bold line with arrowheads at both ends indicates a symmetric link between two nodes, and a regular line with an arrowhead at only one end indicates an asymmetric link between two nodes.

Tables IVa, IVb, and IVc are route tables built in nodes **301, 304,** and **305**, respectively, where valid entries in the *Time_stmp* fields are represented by the symbol *. (The exact values are not critical to this discussion.)

Table IVa (node 301)

Dest_addr	Nxt_hp_addr	Up_str_addr	Hop_cnt	Time_stmp
300	300	305	1	*
306	305	300	3	*

Table IVb (node 304)

Dest_addr	Nxt_hp_addr	Up_str_addr	Hop_cnt	Time_stmp
306	305	303	3	*
303	303	-	1	*

Table IVc (node 305)

Dest_addr	Nxt_hp_addr	Up_str_addr	Hop_cnt	Time_stmp
300	301	302	2	*
306	302	301	2	*
306	302	304	2	*

If node **305** notices the node **302** is unreachable, it informs the related upstream nodes about the broken link, that is, node **305** sends an error packet to nodes **301** and nodes **304** in the concerned routes.

We first consider the case in which a reverse route back to the source node is available. A similar procedure would be followed in the case where there exists an available route which is not the reverse route. In this case, node **305** erases the route entry (6) in Table IVc:

306	302	301	2	*

(6).

Node **305** also sends an error packet (7) to node **301**:

Type	301	305	306

(7)

When node **301** receives the error packet (7), because the values in both the *Nxt_hp_addr* and *Dest_addr* fields in its route table (Table IVa), **305** and **306**, respectively, match the *Src_addr* and *Route_Dest_addr* values in the error packet, node **301** erases the following route entry in Table IVa:

306	305	300	3	*

(8)

Node **301** also sends an error packet (9) to node **300**:

Type	300	301	306

(9)

Finally, the source node **300** receives the error packet (9) and erases the relevant route entry in its route table.

We next consider the case in which no route is available from a current node to an upstream node. In this case, the current node sends an error packet to the upstream node by broadcasting. The format of the broadcast error packet (10) may be as shown below:

Type	Dest_addr	Src_addr	Err_id	Route_Dest_addr

(10)

The *Dest_addr*, *Src_addr*, and *Route_Dest_addr* fields have the same significance in the error packet (10) as in the error packet (5). The *Err_id* field holds a sequence number set by the node corresponding to the *Src_addr*. *Err_id* is combined with *Src_addr* to prevent a node from broadcasting a duplicate error packet. (As a result no node will process the same error packet twice.) The *Type* field indicates that the packet (10) is a broadcasting route error packet.

When the upstream node receives the error packet (10), it erases the route entry in its own route table in which the values in the *Nxt_hp_addr* and *Dest_addr* fields match those in the *Src_addr* and *Route_Dest_addr* fields of the error packet (10), rebuilds the error packet by placing the address of an upstream node address in the *Dest_addr* field, placing its own node address in the *Src_addr* field, and choosing an *Err_id*, and rebroadcasts the rebuilt error packet.

When, in the above example, node **305** notices that the node **302** is unreachable, it erases the following route entry in node **305**:

306	302	304	2	*

(11)

Node 305 also informs its upstream node 304 about the unavailable route to node 306. Since there is no available route to node 304 (the link from node 304 to node 305 being asymmetric), it will broadcast the error packet (12):

Type	304	305	Err_id	306

(12)

The error packet (12) is rebroadcast through nodes 301, 300, and 303 to reach node 304.

When node 304 receives the packet (12), because the *Nxt_hp_addr* and *Dest_addr* values in its route table (Table IVb), 305 and 306, respectively, match the *Src_addr* and *Route_Dest_addr* values in the packet, it erases following route entry in Table IVb:

306	305	303	3	*

(13)

Node 304 also sends an error packet (14) to node 303:

Type	303	304	Err_id	306

(14)

It should be understood that the value of Err_id changes each time the error packet is rebuilt. Finally, the source node 303 receives this error packet and is thereby informed of the defective route.

Figure 4 provides an overview of an exemplary embodiment of the route maintenance mechanism according to the present invention. At 400, a node determines that a next hop node in a route is unreachable, as described previously. At 402, the node determines whether a route is available to an upstream node in the route. If an available route exists at 402, then at 404, the node erases a route entry containing information regarding the failed link, forms an error packet (406), and

unicasts the error packet to an upstream node along the available route (408). The upstream node erases a route entry containing information regarding the failed link (410). The upstream node then determines by examining the error packet whether it is the requesting source node for the route containing the failed link (412). If the upstream node is not the source node, it modifies the error packet, as described previously (414) and unicasts the modified error packet. If the upstream node is the source node, all nodes in the route have been alerted to the failed link, and route maintenance is complete (416).

If an available route does not exist at 402, then the node forms an error packet (418) and broadcasts the error packet (420). If a node receiving the error packet is not the upstream node (422), it rebroadcasts the error packet (420). If the node is the upstream node, it erases a route entry containing information regarding the failed link (424). The upstream node determines by examining the received error packet whether it is the requesting source node for the route containing the failed link (426). If the receiving node is not the source node, it modifies the error packet (428) and broadcasts the modified error packet (420). If the upstream node is the source node, route maintenance is complete (430).

The routing protocol of the present invention has a number of advantages over the prior art. For example, the routing protocol according to the present invention is on-demand in nature and carries all the advantages of on-demand routing protocols compared to table driven protocols, since no periodic route table information exchange is required. The routing protocol of the present invention is also capable of supporting ad hoc networks having asymmetric links. This functionality is achieved without adding packet overhead, as in the case of protocols relying on source routing mechanisms, such as DSR. The present invention requires only minimal routing information in each data packet, as does AODV, but, unlike AODV, works properly in ad hoc networks having asymmetric links. It also provides reverse route formation with explicit link status. In addition, node neighbor discovery eliminates the need to carry out route discovery in the case of destinations that are neighbors. Moreover, since a single pair of RREQ and RREPs may be used,

modified as appropriate by intermediate nodes as the packets propagate through the network, a larger number of nodes are able to derive partial or complete route information, since explicit route and link status are known to the nodes. Furthermore, the route maintenance algorithm of the present invention minimizes the number of stale routes.

Various preferred embodiments of the invention have now been described. While these embodiments have been set forth by way of example, various other embodiments and modifications will be apparent to those skilled in the art. Accordingly, it should be understood that the invention is not limited to such embodiments, but encompasses all that which is described in the following claims.

What is claimed is:

1. A routing protocol for a mobile ad hoc network having a plurality of nodes interconnected by symmetric and/or asymmetric links, each node being associated uniquely with a corresponding route table, the protocol comprising:

discovering a route for a data packet from a first node to a second node;

routing the data packet through the discovered route based on information in each route table along the discovered route; and

maintaining the route.

2. The routing protocol according to claim 1, wherein the first node is a requesting source node and the second node is a neighbor node to the first node, and the discovering a route comprises checking a neighbor list.

3. The routing protocol according to claim 1, wherein the discovering a route comprises:

checking a neighbor list;

broadcasting a first route request packet (RREQ); and

receiving a route reply packet (RREP).

4. The routing protocol according to claim 3, wherein the first RREQ is received and modified by an intermediate node.

5. The routing protocol according to claim 4, wherein the intermediate node checks a status of a flag in the first RREQ and adds a reverse route entry to the route table corresponding to the intermediate node when the flag indicates a reverse route exists to the first node.

6. The routing protocol according to claim 5, wherein the intermediate node broadcasts the modified first RREQ.

7. The routing protocol according to claim 5, wherein the second node forms the RREP and unicasts the RREP to a next hop node along the reverse route.

8. The routing protocol according to claim 7, wherein the next hop node modifies the RREP and builds a forward route entry in the route table corresponding to the reverse route.

9. The routing protocol according to claim 1, wherein the discovering a route comprises:

checking a neighbor list;

broadcasting a first route request packet (RREQ); and

receiving a route reply packet (RREP) including discovered route information.

10. The routing protocol according to claim 1, wherein the first RREQ is received and modified by an intermediate node.

11. The routing protocol according to claim 10, wherein the intermediate node checks a status of a flag in the first RREQ and broadcasts the modified first RREQ when the status of the flag indicates a reverse route to the first node does not exist.

12. The routing protocol according to claim 10, wherein the second node forms the RREP using the modified first RREQ and unicasts the RREP to a next hop node along an available route.

13. The routing protocol according to claim 12, wherein the first node unicasts an activating data packet including the discovered route information to a node along the discovered route after receiving the RREP.

14. The routing protocol according to claim 13, wherein the node builds a forward route entry in the route table corresponding to the discovered route.

15. The routing protocol according to claim 1, wherein the discovering a route comprises:

checking a neighbor list;

broadcasting a first route request packet (RREQ); and

receiving a second RREQ including discovered route information.

16. The routing protocol according to claim 15, wherein the first RREQ is received and modified by an intermediate node.

17. The routing protocol according to claim 16, wherein the intermediate node checks a status of a flag in the first RREQ and broadcasts the modified first RREQ when the status of the flag indicates a reverse route to the first node does not exist.

18. The routing protocol according to claim 16, wherein the second node forms the second RREQ using the modified first RREQ and broadcasts the second RREQ.

19. The routing protocol according to claim 18, wherein the first node unicasts an activating data packet including the discovered route information to a node along the discovered route after receiving the second RREQ.

20. The routing protocol according to claim 19, wherein the node builds a forward route entry in the route table corresponding to the discovered route.

21. The routing protocol according to claim 1, wherein the information includes a route entry.

22. The routing protocol according to claim 21, wherein the route entry includes a timer and the route entry is erased when the timer expires.

23. The routing protocol according to claim 22, wherein the timer is reset when the route entry is used to route the data packet.

24. The routing protocol according to claim 1, wherein the maintaining the discovered route comprises:

determining that a next hop node is unreachable;

transmitting link failure information to at least one upstream node; and

erasing at least one route entry.

25. The routing protocol according to claim 24, wherein the transmitting link failure information comprises unicasting an error packet to the at least one upstream node.

26. The routing protocol according to claim 25, wherein the at least one upstream node modifies the error packet and unicasts the modified error packet.

27. The routing protocol according to claim 26, wherein the transmitting link failure information comprises broadcasting an error packet.

28. The routing protocol according to claim 27, wherein the at least one upstream node modifies the error packet and broadcasts the modified error packet.

29. The routing protocol according to claim 24, wherein the at least one upstream node erases the at least one route entry.

Figure 1a

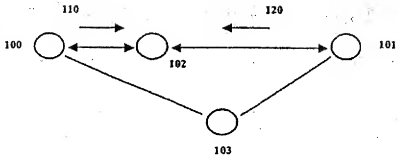
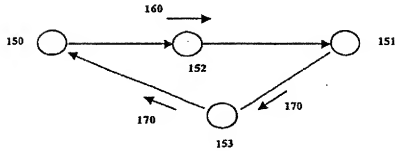


Figure 1b



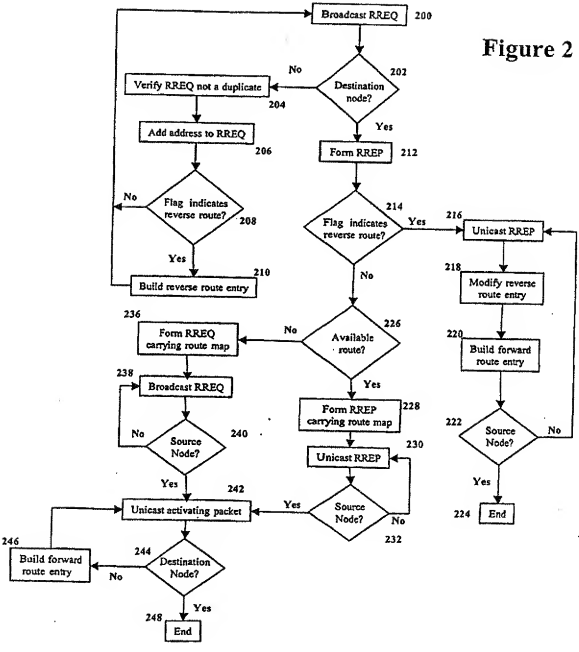

Broadcast RREQ
200
Destination node?
202
No
Verify RREQ not a duplicate
204
Add address to RREQ
206
Flag indicates reverse route?
208
No
Yes
Build reverse route entry
210
Yes
Form RREP
212
Flag indicates reverse route?
214
Yes
Unicast RREP
216
Modify reverse route entry
218
Build forward route entry
220
Source Node?
222
No
Yes
End
224
No
Available route?
226
Form RREP carrying route map
228
Unicast RREP
230
Source Node?
232
No
Yes
Form RREQ carrying route map
236
Broadcast RREQ
238
Source Node?
240
No
Yes
Unicast activating packet
242
Destination Node?
244
No
Build forward route entry
246
Yes
End
248

Figure 2

Figure 3

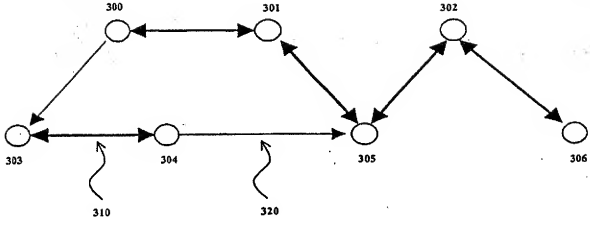
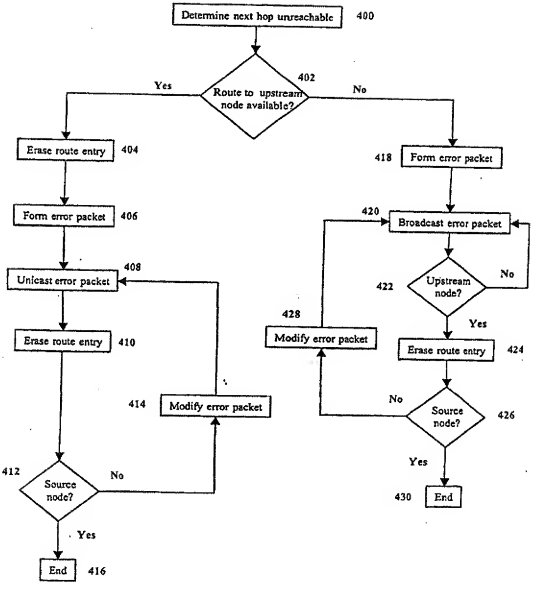


Figure 4


Determine next hop unreachable
400
Route to upstream node available?
402
Yes
No
Erase route entry
404
Form error packet
406
Unicast error packet
408
Erase route entry
410
Source node?
412
No
Yes
End
416
Modify error packet
414
Form error packet
418
Broadcast error packet
420
Upstream node?
422
No
Yes
Erase route entry
424
Source node?
426
No
Yes
End
430
Modify error packet
428

INTERNATIONAL SEARCH REPORT

International application No.
PCT/SG 01/00064

CLASSIFICATION OF SUBJECT MATTER

IPC[7]: H04L 12/56, H04L 29/06, H04Q 7/38

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

IPC[7]: H04L, H04Q

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

WPI, EPODOC

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 5987011 A (TOH, C.K.) 16 November 1999 (16.11.99) *abstract, figs. 5b,6a,6b,6c,8d; column 7, line 65 - column 11, line 65; claims.*	1-4,9,10,12
X	RAMANUJAN, R. et al. 'Source-Initiated Adaptive Routing Algorithm (SARA) for Autonomous Wireless Local Area Networks.' In: Proceedings of the 23rd Annual Conference on Local Computer Networks. New York: IEEE, 1998. ISBN 0-8186-8810-6, pages 109 to 118, *especially pages 111 to 115.*	1-6,9,10,15, 16,18,21,22
X	PERKINS, C. et al. 'Ad-hoc On-Demand Distance Vector Routing.' In: Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications. New York: IEEE, 1999. ISBN 0-7695-0025-0, pages 90 to 100, *especially pages 91 to 94.*	1-3,21-24,29

☒ Further documents are listed in the continuation of Box C. ☒ See patent family annex.

* Special categories of cited documents:
„A" document defining the general state of the art which is not considered to be of particular relevance
„E" earlier application or patent but published on or after the international filing date
„L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
„O" document referring to an oral disclosure, use, exhibition or other means
„P" document published prior to the international filing date but later than the priority date claimed

„T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
„X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
„Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
„&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
15 January 2002 (15.01.2002)	28 February 2002 (28.02.2002)
Name and mailing adress of the ISA/AT Austrian Patent Office Kohlmarkt 8-10; A-1014 Vienna Facsimile No. 1/53424/535	Authorized officer LOIBNER Telephone No. 1/53424/323

Form PCT/ISA/210 (second sheet) (July 1998)

INTERNATIONAL SEARCH REPORT

International application No.
PCT/SG 01/00064

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	DONKYUN, K. et al. 'RODA: A new dynamic routing protocol using dual paths to support assymmetric links in mobile ad hoc networks.' In: Proceedings of the 9th International Conference on Computer Communications and Networks. New York: IEEE, 2000. ISBN 0-7803-6494-5, pages 4 to 8, *especially figures 1 to 4 and pages 5 to 7.*	1,24,25

Form PCT/ISA/210 (continuation of second sheet) (July 1998)

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.
PCT/SG 01/00064

Patent document cited in search report			Publication date	Patent family member(s)	Publication date
RA	A	M		none	
US	A	5987011	16-11-1999	none	